最小消去多項式候補とその応用 (Computer Algebra : Design of Algorithms, Implementations and Applications)

AUTHOR(S):

田島, 慎一; 奈良, 洸平

数理解析研究所講究録
第1815巻 2012年 1-12

最小消去多項式候補とその応用

田島慎一
筑波大学大学院 数理物質科学研究科*
Shinichi Tajima
Graduate School of Pure and Applied Sciences, University of Tsukuba

奈良洸平
新潟大学大学院 自然科学研究科
Kohei Nara
Graduate School of Science and Technology, Niigata University

1 序

本稿では, 最小消去多項式候補を求める算法と最小消去多項式候補を用いた最小多項式計算について論じる. 本論に入る前に先ずこの節で, 論文 [9] においてスペクトル分解行列の計算法を導出した際の基本的考え方を紹介し, これにより最小消去多項式候補を効率的に求める新たな算法の必要性を明らかにしておく.

一般に, 最小多項式（あるいは特性多項式）を用いることで, 行列 A のレゾルベント $(\lambda A - E)^{-1}$ を（最小多項式あるいは特性多項式を共通の分母として持つような有理関数係数の）行列多項式として表現する式を導くことができる. 関数解析学におけるスペクトル分解定理をレゾルベントの行列多項式による表現に適用することで, 与えられた（整数あるいは有理数を成分に持つ）正方行列のスペクトル分解行列を求めるアルゴリズムを構成できる. この方法を特性多項式が既約でない行列に対して適用する際は, 各既約因子毎に計算を行うことで計算の効率化を図れる. しかし, 最小多項式（あるいは特性多項式）から導いたレゾルベントの行列多項式表現は, 注目した既約因子に対応するスペクトル分解行列の計算に用いる表現式としては, 冗長であることが分かる. 従って, 特性多項式が既約でない行列のスペクトル分解を求める計算法の導出に際しては, まず, 既約因子毎に計算に適した行列多項式表現を構成し, さらにこれらの表現式に基づくことで既約因子に対応するスペクトル分解分解行列を効率よく計算するアルゴリズムを設計することが必要となる.

*tajima@math.tsukuba.ac.jp

さて, いま, 行列 A は $n \times n$ 正方行列, $\mathbf{e}_j$ $(1 \leq j \leq n)$ は, 第 j 成分のみ 1 に等しく他の成分は全て零であるような n 次元基本単位ベクトルとし, 行列 A の基本ベクトル $\mathbf{e}_j$ に関する最小消去多項式を基本最小消去多項式と呼び, $\pi_{A,j}(\lambda)$ で表す. 論文 [9] では, これら n 個の基本最小消去多項式を用いることで, 行列 A の特性多項式の既約因子に関する最小消去多項式とも称すべき概念を導入した. さらに, この概念を用いることで各既約因子に対応するスペクトル分解行列の計算に適した (レゾルベントの) 行列多項式表現を導出し, これによりスペクトル分解計算の効率化を図れることを示した.

論文 [9] で提案した方法に基づいて実際に計算アルゴリズムを設計するには, n 個の基本最小消去多項式 $\pi_{A,j}(\lambda)$, $j = 1, 2, ..., n$ を効率良く求める新たな算法を考案することが重要となる. 本稿ではまず, [9] に従って, これらの基本最小消去多項式 $\pi_{A,j}(\lambda)$, $j = 1, 2, ..., n$ を求める際に要となる基本最小消去多項式候補について説明し, 次に試作したプログラムの性能を評価する. 最後に応用として基本最小消去多項式候補を利用した最小多項式計算について論じる.

2 最小消去多項式候補計算の基礎

まず, 最小消去多項式の概念について復習することからはじめる. 整数を成分に持つ $n \times n$ 正方行列 A と n 次元ベクトル $\boldsymbol{v}$ に対し, 有理数係数の一変数多項式全体のなす環 $K[\lambda]$ におけるイデアル

$$Ann_{K[\lambda]}(A, \boldsymbol{v}) := \{p(\lambda) \in K[\lambda] \mid p(A)\boldsymbol{v} = 0\}$$

の monic な生成元を, 行列 A のベクトル $\boldsymbol{v}$ に関する最小消去多項式と呼ぶ. 特に, 第 j 成分のみ 1 に等しく他の成分は全て零であるような n 次元基本単位ベクトル $\mathbf{e}_j$, $(1 \leq j \leq n)$ に関する最小消去多項式を $\pi_{A,j}(\lambda)$ で表し, 基本最小消去多項式と呼ぶことにする.

ベクトル $\boldsymbol{v}$ が与えられた時, この一つのベクトルに関する最小消去多項式を求めることは比較的容易である. しかし, n 個の基本最小消去多項式を全て求める必要があるとき各 $\pi_{A,j}(\lambda)$ を一つ一つ個別に求めていたのでは, 行列サイズが大きい場合, かなりの計算量となる. 従って, 既存の計算法をそのままの形で利用したのでは, n 個の基本最小消去多項式 $\pi_{A,j}(\lambda), j = 1, 2, ..., n$ を求める算法としては実用性に欠けることは明らかである. そこで, 基本最小消去多項式の代わりに, その候補に着目し, これらを全て同時に求めるような計算法を考える. 今, n 次元行ベクトル $\boldsymbol{u}$ に対し, 多項式 $p(\lambda)$ に行列 A を代入した行列多項式 $p(A)$ を右から施すことで得られる行ベクトル $\boldsymbol{w}=\boldsymbol{u}p(A)$ を考える. ベクトル $\boldsymbol{w}$ の成分表示を $\boldsymbol{w}= (w_1, w_2, ..., w_j, ...w_n)$ とすると $\boldsymbol{u}(p(A)\mathbf{e}_j) = (\boldsymbol{u}p(A))\mathbf{e}_j = \boldsymbol{w}\mathbf{e}_j = w_j$ より $p(A)\mathbf{e}_j = 0$ ならば $w_j = 0$ となることは明らかである. 従って, n 次元ベクトル $\boldsymbol{w}$ の第 j 成分が零でなければ, n 次元ベクトル $p(A)\mathbf{e}_j$ は零ベクトルでないことが, すべての j について成り立つことになる. 以下に示すように, 論文 [9] で与えた最小消去多項式候補の計算法はこの事実に基づいている.

さて, 行列 A の特性多項式 $\chi_A(\lambda)$ の因数分解を

$$\chi_A(\lambda) = f_1(\lambda)^{m_1} f_2(\lambda)^{m_2} \cdots f_q(\lambda)^{m_q},$$

各 $j \in \{1,2,...,n\}$ に関する最小消去多項式の因数分解を

$$\pi_{A,j}(\lambda) = f_1(\lambda)^{r_{j,1}} f_2(\lambda)^{r_{j,2}} \cdots f_q(\lambda)^{r_{j,q}}$$

とする. いま, $1 \leq p \leq q$ なる p に対し, 特性多項式 $\chi_A(\lambda)$ の因子 $f_p(\lambda)$ 以外の因子の積

$$g_p(\lambda) = f_1(\lambda)^{m_1} f_2(\lambda)^{m_2} \cdots f_{p-1}(\lambda)^{m_{p-1}} f_{p+1}(\lambda)^{m_{p+1}} \cdots f_q(\lambda)^{m_q}$$

が定める行列 $g_p(A)$ を G_p で表す. このとき, 最小消去多項式

$$\pi_{A,j}(\lambda) = f_1(\lambda)^{r_{j,1}} f_2(\lambda)^{r_{j,2}} \cdots f_q(\lambda)^{r_{j,q}}$$

の因子 $f_p(\lambda)$ の指数 $r_{j,p}$ は, $F_p = f_p(A)$ とおくと, $F_p^k G_p \mathbf{e}_j = 0$ となる最小の自然数 k として特徴付けることができる. いま, n 次元行ベクトル $\boldsymbol{u}$ に行列 G_p を右から施して得られる行ベクトル $\boldsymbol{u}G_p$ を $\boldsymbol{w}_p^{(0)}$ で表し, その成分表示を $\boldsymbol{w}_p^{(0)} = (w_{p,1}^{(0)}, w_{p,2}^{(0)}, ..., w_{p,n}^{(0)})$ とおく. 同様に, n 次元行ベクトル $\boldsymbol{u}G_pF_p^k$ を $\boldsymbol{w}_p^{(k)} = (w_{p,1}^{(k)}, w_{p,2}^{(k)}, ..., w_{p,n}^{(k)})$ で表す. これらを用いて自然数 $\rho_{p,j},\ j = 1,2,...,n$ を次のように定める.

- $w_{p,j}^{(0)} = 0$ ならば $\rho_{p,j} = 0$
- $w_{p,j}^{(k-1)} \neq 0, w_{p,j}^{(k)} = 0$ ならば $\rho_{p,j} = k$

先程述べたことを $p(\lambda) = f_p(\lambda)^k g_p(\lambda)$ に対し適用することで次を得る.

補題 ([9]) $r_{j,p} \geq \rho_{p,j}$ が成り立つ.

従って, 各 $p = 1,2,...,q$ 毎に $\rho_{p,j}, j = 1,2,..,n$ を求め

$$\pi'_{A,j}(\lambda) = f_1(\lambda)^{\rho_{1,j}} f_2(\lambda)^{\rho_{2,j}} \cdots f_q(\lambda)^{\rho_{q,j}}$$

と定めれば, 多項式 $\pi'_{A,j}(\lambda)$ は最小消去多項式 $\pi_{A,j}(\lambda)$ を割り切る. 本稿では, この様にして得られる多項式 $\pi'_{A,j}(\lambda), j = 1,2,...,n$ のことを単に基本最小消去多項式候補とよぶことにする.

さて, ここに述べた方法で基本最小消去多項式候補 $\pi'_{A,j}(\lambda), j = 1,2,...,n$ を求めるには, まず, q 個の n 次元ベクトル

$$\boldsymbol{w}_1^{(0)} = \boldsymbol{u}G_1, \boldsymbol{w}_2^{(0)} = \boldsymbol{u}G_2, ..., \boldsymbol{w}_q^{(0)} = \boldsymbol{u}G_q$$

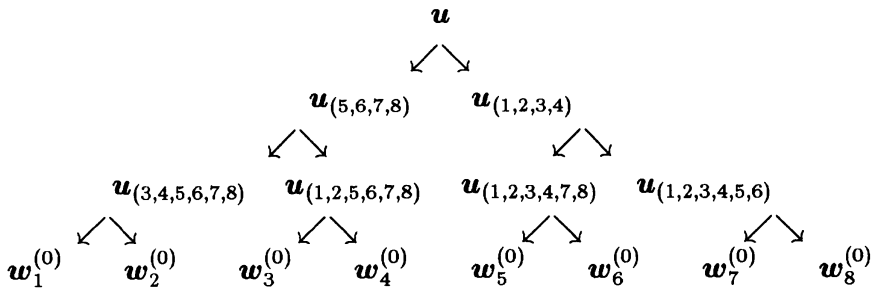
を求めておく必要がある. 定義より, $\boldsymbol{w}_1^{(0)}, \boldsymbol{w}_2^{(0)}, ..., \boldsymbol{w}_q^{(0)}$ は

$$\begin{aligned}
\boldsymbol{w}_1^{(0)} &= \boldsymbol{u}F_2^{m_2}F_3^{m_3}\cdots F_q^{m_q} \\
\boldsymbol{w}_2^{(0)} &= \boldsymbol{u}F_1^{m_1}F_3^{m_3}\cdots F_q^{m_q} \\
\cdot &= \cdots \\
\cdot &= \cdots \\
\cdot &= \cdots \\
\boldsymbol{w}_q^{(0)} &= \boldsymbol{u}F_1^{m_1}F_2^{m_2}\cdots F_{q-1}^{m_{q-1}}
\end{aligned}$$

により与えられる. 行ベクトル $\boldsymbol{u}$ に施す行列は, A の行列多項式として, 互いに殆ど同じ因子からなる. 従って, [9] で述べたように, 二分木法を用いることで $\boldsymbol{w}_1^{(0)}, \boldsymbol{w}_2^{(0)}, ..., \boldsymbol{w}_q^{(0)}$ を求める際の計算量を軽減できる.

簡単のため, $q = 8$ の場合を例にとり, 二分木を用いた $\boldsymbol{w}_1^{(0)}, \boldsymbol{w}_2^{(0)}, ..., \boldsymbol{w}_8^{(0)}$ の求め方を説明する. 以下, ベクトル $\boldsymbol{u}F_{i_1}^{m_{i_1}}F_{i_2}^{m_{i_2}}\cdots F_{i_k}^{m_{i_k}}$ を $\boldsymbol{u}_{(1_1,1_2,...,1_k)}$ と略記する. この記号を用いると, 例えば $\boldsymbol{w}_1^{(0)}$ は, いまの場合 $\boldsymbol{w}_1^{(0)} = \boldsymbol{u}_{(2,3,4,...,8)}$ と表せる. まずはじめに, ベクトル $\boldsymbol{u}$ より $\boldsymbol{u}_{(5,6,7,8)} = \boldsymbol{u}F_5^{m_5}F_6^{m_6}F_7^{m_7}F_8^{m_8}$ を求める. 次に, このベクトル $\boldsymbol{u}_{(5,6,7,8)}$ に $F_3^{m_3}F_4^{m_4}$ と $F_1^{m_1}F_2^{m_2}$ を右から施すことで, $\boldsymbol{u}_{(3,4,5,6,7,8)}$, $\boldsymbol{u}_{(1,2,5,6,7,8)}$ を求める. 更に, 前者に $F_2^{m_2}$ と $F_1^{m_1}$, 後者に $F_4^{m_4}$ と $F_3^{m_3}$ を右から施すことで, $\boldsymbol{u}_{(2,3,4,5,6,7,8)}$, $\boldsymbol{u}_{(1,3,4,5,6,7,8)}$, $\boldsymbol{u}_{(1,2,4,5,6,7,8)}$, $\boldsymbol{u}_{(1,2,3,5,6,7,8)}$ 即ち, $\boldsymbol{w}_1^{(0)}$, $\boldsymbol{w}_2^{(0)}$, $\boldsymbol{w}_3^{(0)}$, $\boldsymbol{w}_4^{(0)}$ を得る. ベクトル $\boldsymbol{u}_{(1,2,3,4)} = \boldsymbol{u}F_1^{m_1}F_2^{m_2}F_3^{m_3}F_4^{m_4}$ に対し先程と同様な計算をすることで, $\boldsymbol{w}_5^{(0)}$, $\boldsymbol{w}_6^{(0)}$, $\boldsymbol{w}_7^{(0)}$, $\boldsymbol{w}_8^{(0)}$ を求められる.

この計算の流れは次のような二分木を用いて図式化できる.



注意 ベクトル $\boldsymbol{u}$ より $\boldsymbol{w}_1^{(0)}, \boldsymbol{w}_2^{(0)}, ..., \boldsymbol{w}_q^{(0)}$ を求める二分木計算も, $\boldsymbol{w}_1^{(0)}, \boldsymbol{w}_2^{(0)}, ..., \boldsymbol{w}_q^{(0)}$ より基本最小消去多項式候補の各因子の重複度 $\rho_{p,j}, p = 1, 2, ..., q, j = 1, 2, ..., n$ を求める計算も共に並列化することが可能である.

3 基本プログラムの試作・実装

本研究では, 最小消去多項式候補を求めるプログラムを4つ試作した. これらのプログラムの原型となるアルゴリズムの概略を以下に与える. 入力は, 整数を成分とする n 次正方行列 A, 行列 A の特性多項式の因数分解からなる.

3.1 行列の最小消去多項式候補の計算アルゴリズム (アルゴリズム I)

STEP 1 ランダムベクトルの生成

・ランダムな整数を成分とする n 次元行ベクトル $\boldsymbol{u}$ を生成する.

STEP 2 二分木による $\boldsymbol{w}_1^{(0)}, \boldsymbol{w}_2^{(0)}, ..., \boldsymbol{w}_q^{(0)}$ の計算

・$X \leftarrow [\boldsymbol{u}]$ (リスト)

・$left \leftarrow 0$, $right \leftarrow$ 特性多項式の因子数 q

・for

$\boldsymbol{x}_1, \boldsymbol{x}_2 \leftarrow$ リスト X の先頭項（最初は $\boldsymbol{u}$）

$pt \leftarrow 1$

$s \leftarrow right - left$

for

if s/pt の商が1もしくは2で余りが0

$key \leftarrow right - pt$

key の位置をメモリに格納してループを抜ける

else

$pt \leftarrow pt \cdot 2$

for $i \leftarrow left$ to $key - 1$

$\boldsymbol{x}_1 \leftarrow \boldsymbol{x}_1 F_i^{m_i}$

for $i \leftarrow key$ to $right - 1$

$\boldsymbol{x}_2 \leftarrow \boldsymbol{x}_2 F_i^{m_i}$

リスト X の先頭項を除き，X へ $\boldsymbol{x}_1, \boldsymbol{x}_2$ を順に左から格納

この時点で $key - left$ が1なら $left \leftarrow left + 1$ とし

X の先頭項を除く

$left = right$ になったらループを抜ける

STEP 3 基本最小消去多項式候補の計算

・for $p \leftarrow q$(特性多項式の因子数) to 1

$(\rho_{p,1}, \rho_{p,2}, ..., \rho_{p,n}) \leftarrow (0, 0, ..., 0)$

$\boldsymbol{w}_p \leftarrow \boldsymbol{w}_p^{(0)}$

for $k \leftarrow 0$ to m_p

for $j \leftarrow n$(行列の列数) to 1

if $w_{p,j} \neq 0, \rho_{p,j} \leftarrow \rho_{p,j} + 1$

この時点で $\boldsymbol{w}_p = \boldsymbol{0}$ (零ベクトル) ならループを抜ける

$\boldsymbol{w}_p \leftarrow \boldsymbol{w}_p F_p$

以上がアルゴリズムIの概略である．計算のほとんどは，ベクトル・行列多項式積からなる．ベクトル・行列多項式積の計算を通常のホーナー法で行うとすると，Step 2 では $n \log_2 q$ 回のベクトル・行列積計算を行い，Step 3 では，最小多項式 $\pi_A(\lambda)$ の因数分解を

$$\pi_A(\lambda) = f_1(\lambda)^{\ell_1} f_2(\lambda)^{\ell_2} \cdots f_q(\lambda)^{\ell_q}$$

とおくと，$\ell_1 \deg f_1 + \ell_2 \deg f_2 + \cdots + \ell_q \deg f_q$ 即ち，$\deg \pi_A$ 回のベクトル・行列積を行う．合計すると，$n \log_2 q + \deg \pi_A$ 回となる．

3.2 アルゴリズムの信頼性向上と効率化

アルゴリズムIの信頼性向上を図るため，ひとつのランダムベクトルではなく，2つのランダムベクトルからなる2 × n行列を使用して最小消去多項式を求めるアルゴリズムIIを導出した．より確実に最小消去多項式が求められることになる．また効率化を図るため，アルゴリズムIの計算を素数を法として行うアルゴリズム I_{mod} を導出した．このアルゴリズムでは，先ず素数を与え，与えられた行列の各要素，特性多項式の各因子の係数等の剰余をとり，与えられた素数による剰余体で計算を進めていく．これにより大きな数値の計算の際に計算コストの軽減が期待できる．アルゴリズム I_{mod} と同様に，2 × n行列を使用し素数を法として計算するアルゴリズム II_{mod} を導出した．

3.3 時間計測実験

今回導出した4つのアルゴリズムのプログラムを数式処理システム Risa/Asir に実装し，CPU 時間を測定，比較し性能を評価した．使用した計算機の OS は，Windows Vista Ultimate，CPU は，Intel Core2 Quad CPU Q9550 @2.83GHz，メモリーは，8.00GB.

3.3.1 実験1（行列のサイズを変えて実験）

測定方法

- 特性多項式の因子の次数は4で固定．その係数は，-1024～1024 の範囲の整数（最高次を除く）．異なる因子の個数は4で固定
- 行列のサイズを 80，160，240，320（特性多項式の因子の重複度はそれぞれ 5，10，15，20）と変えてそれぞれ測定
- 最小多項式の因子の重複度は特性多項式の因子の重複度に近い範囲に設定
- 測定に使用する行列は，要素が -1024～1024 の範囲の行列を 10 回作成してビット長の平均を取り，それとのビット長の差が ±10 %となるように作成
- アルゴリズム I_{mod}，アルゴリズム II_{mod} で使用する素数は 1229 とする
- それぞれのプログラムで2回ずつ測定し，CPU 時間の平均を算出

測定結果

測定結果を以下に示す．なお，表1はアルゴリズムの前半部分（STEP 1, 2），表2は最小消去多項式候補の計算部分（STEP 3）のデータである．

表１：実験結果（二分木計算までの時間）

サイズ	CPU 時間 (sec)			
	プログラム I	プログラム I_{mod}	プログラム II	プログラム II_{mod}
８０	0.2	0.2	0.4	0.3
１６０	2.5	1.3	4.9	2.5
２４０	10.7	5.0	20.8	9.6
３２０	24.9	10.2	49.1	19.5

表２：実験結果（最小消去多項式候補の計算時間）

サイズ	CPU 時間 (sec)			
	プログラム I	プログラム I_{mod}	プログラム II	プログラム II_{mod}
８０	0.1	0.1	0.1	0.1
１６０	1.0	0.5	1.9	1.0
２４０	4.7	2.2	9.1	4.1
３２０	13.1	4.6	25.2	8.9

3.3.2 実験２（最小消去多項式の総次数を変えて測定）

測定方法

- 行列のサイズは 320 で固定
- 特性多項式の因子の次数は４で固定．各因子の係数は，-1024～1024 の範囲の整数（最高次を除く.）因子の重複度は全て 20 で固定
- 測定に使用する行列は，要素が -1024～1024 の範囲の行列を 10 回作成してビット長の平均を取り，それとのビット長の差が ±10 %となるように作成
- アルゴリズム I_{mod}，アルゴリズム II_{mod} で使用する素数は 1229 とする
- 最小多項式の因子の重複度を変えてそれぞれ測定
- それぞれのプログラムで２回ずつ測定し，CPU 時間の平均を算出

測定結果

測定結果を以下に示す．なお，表１はアルゴリズムの前半部分（STEP 1, 2），表２は最小消去多項式候補の計算部分（STEP 3）のデータである．

表３：実験結果（二分木計算までの時間）

因子の重複度	CPU 時間 (sec)			
	プログラム I	プログラム I_{mod}	プログラム II	プログラム II_{mod}
１～４	24.5	9.8	48.2	18.9
９～１２	23.4	9.8	45.7	19.4
１７～２０	24.9	10.2	49.1	19.5

表４：実験結果（最小消去多項式候補の計算時間）

因子の重複度	CPU 時間 (sec)			
	プログラム I	プログラム I_{mod}	プログラム II	プログラム II_{mod}
１～４	2.4	0.6	3.0	1.1
９～１２	6.4	2.5	11.7	4.9
１７～２０	13.1	4.6	25.2	8.9

実験１の結果, 行列サイズ小さい場合は素数を法とした剰余計算の効果はほとんど現れないが, 行列サイズが増すにつれ剰余計算の効果が計算所要時間に顕著に顕れることを確認できた. また, 実験２の結果, Step 2 での計算量は最小多項式の次数に依らず一定であるが, Step 3 での計算量は, 最小多項式の次数に依存することを確かめることができた.

4 改良の試み

第３節で与えたアルゴリズムは, 行列 A の特性多項式 $\chi_A(\lambda)$ の次数に比べ最小多項式 $\pi_A(\lambda)$ の次数が小さい場合, Step 2 において結果的にはかなり無駄の多い計算を行うことになる. 実際, 行列 A の特性多項式 $\chi_A(\lambda) = f_1(\lambda)^{m_1} f_2(\lambda)^{m_2} \cdots f_q(\lambda)^{m_q}$ に対し, 最小多項式 $\pi_A(\lambda)$ の因数分解を $\pi_A(\lambda) = f_1(\lambda)^{\ell_1} f_2(\lambda)^{\ell_2} \cdots f_q(\lambda)^{\ell_q}$ とすると, 明らかに, $1 \leq \ell_p \leq m_p, p = 1, 2, ..., q$ であり, さらに ℓ_p は各 $j \in \{1, 2, ..., n\}$ に関する最小消去多項式

$$\pi_{A,j}(\lambda) = f_1(\lambda)^{r_{j,1}} f_2(\lambda)^{r_{j,2}} \cdots f_q(\lambda)^{r_{j,q}}$$

の因子 f_p の重複度により,

$$\ell_p = \max\{r_{1,p}, r_{2,p}, ..., r_{n,p}\}$$

と表せる. 従って, これらのことから, もし仮に今, 行列の最小多項式が既知であるとし, 特性多項式ではなく最小多項式を用いて基本最小消去多項式候補を求めるアルゴリズム I を動かしたとすると Step 2 でのベクトル・行列積計算は $\deg \pi_A(\log_2 q)$ まで減らせることが分かる. つまり, アルゴリズム I では, Step 2 において特性多項式を用いるため, 理論的にはベクトル・行列積を $(n - \deg \pi_A) \log_2 q$ 回ほど余分な計算をしていることになる.

この種の問題では, 行列の最小多項式が予め分かっていることは期待できない. そこで, 以下に,「最小多項式候補を用いた基本最小消去多項式候補計算の改良」を提案する.

4.1 最小多項式候補の作成とその利用

先ず, 二つの n 次元ランダムベクトル $\boldsymbol{u}$, $\boldsymbol{v}$ を生成する. ただし $\boldsymbol{u}$ は行ベクトル, $\boldsymbol{v}$ は列ベクトルとする. 行列 $\Gamma = F_1F_2\cdots F_q$ を $\boldsymbol{v}$ に左から施して得られる列ベクトルを $\boldsymbol{v}_\Gamma$ とおく. $\boldsymbol{v}_\Gamma = 0$ なる場合, $\pi'_A(\lambda) = f_1(\lambda)f_2(\lambda)\cdots f_q(\lambda)$ を最小多項式候補と定める.

$\boldsymbol{v}_\Gamma \neq 0$ なる場合, このベクトルをメモリーに保存し, さらにベクトル $\boldsymbol{v}_\Gamma$ に左から F_q を施す. 計算結果が零ベクトルでない限りこれを $m_q - 1$ 回繰り返し, $F_q^{m_q-1}\boldsymbol{v}_\Gamma$ を求め, メモリーに保存する. つぎに, ベクトル $F_q^{m_q-1}\boldsymbol{v}_\Gamma$ に左から, F_{q-1} を施していく. 先程と同様に, 零ベクトルが得られない限りこれを繰り返し $F_{q-1}^{m_{q-1}-1}(F_q^{m_q-1}\boldsymbol{v}_\Gamma)$ を求め, これをメモリーに保存する. 以下, 同様の計算を繰り返す.

いま, メモリーに, $F_{s+1}^{m_{s+1}-1}F_{s+2}^{m_{s+2}-1}\cdots F_q^{m_q-1}\boldsymbol{v}_\Gamma$ までのベクトルが保存されているとし,

$$F_s^{k-1}(F_{s+1}^{m_{s+1}-1}F_{s+2}^{m_{s+2}-1}\cdots F_q^{m_q-1}\boldsymbol{v}_\Gamma) \neq 0, F_s^k(F_{s+1}^{m_{s+1}-1}F_{s+2}^{m_{s+2}-1}\cdots F_q^{m_q-1}\boldsymbol{v}_\Gamma) = 0$$

となったとする. この時,

$$\ell'_1 = \ell'_2 = \cdots = \ell'_{s-1} = 1, \ell'_s = k+1$$

と定める. さらに, 行ベクトル $\boldsymbol{u}$ に右から F_s^k を施して得られる行ベクトル $\boldsymbol{u}F_s^k$ を求め, このベクトルを再び, $\boldsymbol{u}$ で表す. 積 $\boldsymbol{u}\cdot F_{s+2}^{m_{s+2}-1}\cdots F_q^{m_q-1}\boldsymbol{v}_\Gamma$ を計算し, この値が零である場合は, $\ell'_{s+1} = 1$ と定める. 積の値が零でない場合は, 行ベクトル $\boldsymbol{u}$ に右から F_{s+1} を施し, 積 $(\boldsymbol{u}F_{s+1})\cdot F_{s+2}^{m_{s+2}-1}\cdots F_q^{m_q-1}\boldsymbol{v}_\Gamma$ を計算する. 以下同様の操作を繰り返し, 積 $(\boldsymbol{u}F_{s+1}^k)\cdot F_{s+2}^{m_{s+2}-1}\cdots F_q^{m_q-1}\boldsymbol{v}_\Gamma$ が初めて零となる k を用いて, $\ell'_{s+1} = k+1$ と定め, $\boldsymbol{u}F_{s+1}^k$ を再び, $\boldsymbol{u}$ とおく. この操作を繰り返すことで, 整数の組 $\ell'_1, \ell'_2, ..., \ell'_q$ を求め, 最小多項式候補

$$\pi'_A(\lambda) = f_1(\lambda)^{\ell'_1}f_2(\lambda)^{\ell'_2}\cdots f_q(\lambda)^{\ell'_q}$$

を構成する. (ベクトル $\boldsymbol{u}$, $\boldsymbol{v}$ をランダムに生成するので, ほとんどの場合, $\pi_A(\lambda) = \pi'_A(\lambda)$ となることが期待できる.)

さて, ここでアルゴリズム I の Step 1 に従い, ランダムな整数を成分に持つ n 次元行ベクトル $\boldsymbol{u}$ をあらたに生成する. この行ベクトルと最小多項式候補 $\pi'_A(\lambda)$ を用いて, Step 2 の手順に従って, $\boldsymbol{w}_1^{(0)}, \boldsymbol{w}_2^{(0)}, ..., \boldsymbol{w}_q^{(0)}$ に相当する q 個の行ベクトル $\boldsymbol{w}_1'^{(0)}, \boldsymbol{w}_2'^{(0)}, ..., \boldsymbol{w}_q'^{(0)}$ を求める. ただし, $\boldsymbol{w}_p'^{(0)}$ は

$$\boldsymbol{w}_p'^{(0)} = \boldsymbol{u}F_1^{\ell'_1}F_2^{\ell'_2}\cdots F_{p-1}^{\ell'_{p-1}}F_{p+1}^{\ell'_{p+1}}\cdots F_q^{\ell'_q}$$

である. これらのベクトルを用いて, Step 3 と同様の計算を行うことで, 基本最小消去多項式候補の計算を行う. これにより, 最小多項式候補が最小多項式と一致する場合, n 個の基本最小消去多項式候補を全て構成することが出来る.

最小多項式候補を構成する際のベクトル・行列多項式積を通常のホーナー法で行うとするとし, ベクトル・行列積の回数を見積もると, ほぼ $n + \deg\pi_A$ 回となる. 従って, $(n - \deg\pi_A)\log_2 q - (n + \deg\pi_A) > 0$ 即ち,

$$\deg\pi_A < \frac{\log_2 q - 1}{\log_2 q + 1}n$$

であれば, もともとのアルゴリズムIより, この節で与えた計算法の方が計算効率が良くなる.

4.2 再計算の仕方

いま, 最小多項式候補の因子 f_p の重複度 ℓ'_p が真の最小多項式の因子 f_p の重複度 ℓ_p と一致せず, $\ell'_p < \ell_p$ であるとする. いま簡単のため, それ以外の因子に対する重複度はすべて一致するとして議論をすすめる（一般化は容易）. 真の基本最小消去多項式

$$\pi_{A,j}(\lambda) = f_1(\lambda)^{r_{j,1}} f_2(\lambda)^{r_{j,2}} \cdots f_q(\lambda)^{r_{j,q}}$$

の因子 f_p の重複度 $r_{j,p}$ が $r_{j,p} \leq \ell'_p$ を満たす j に対しては, 上記の方法で, 基本最小消去多項式候補を構成することができる. それに対し, $r_{j,p} > \ell'_p$ なる j に関しては, 上記の計算によりほとんどすべての $s = 1, 2, ..., q$ に対しほぼ間違いなく $w_{s,j}^{\prime(\ell'_s)} \neq 0$ を得ることになり, 基本最小消去多項式候補を構成できない.

そこで, 第3節で与えたアルゴリズムIの Step 3 の計算を最小多項式候補 $\pi'_A(\lambda)$ を用いて行った場合に得られるベクトル $\boldsymbol{w}_s^{\prime(\ell'_s)}, s = 1, 2, ..., q$ を用いて, 集合

$$J' = \{j \mid 1 \leq j \leq n, \exists s, w_{s,j}^{(\ell'_s)} \neq 0\}$$

を導入する. $J' = \emptyset$ なる場合は, すべての $\{1, 2, ..., n\}$ に対し, 基本最小消去多項式候補を構成できたことになる. $J' \neq \emptyset$ なる場合, 明らかに, $j \in J'$ なる j に対する基本最小消去多項式 $\pi_{A,j}(\lambda)$ の少なくとも一つの因子 f_s の重複度 $r_{j,s}$ が最小多項式候補 $\pi'_A(\lambda)$ のそれより大きいことになる.

以上の考察に基づいて, $J' \neq \emptyset$ なる場合, 次のように計算を行うことを提案する.

- 先ず, 列ベクトル $\boldsymbol{v}'$ であり, 各 $j \in J'$ 成分はランダムに生成した整数, $j \notin J'$ 成分は全て零に等しいような n 次元ベクトルを作る. この列ベクトルの最小消去多項式 $\pi_{A,\boldsymbol{v}'}(\lambda)$ を求める.
- ランダム行ベクトル $\boldsymbol{u}'$ を生成. 今度は, 最小消去多項式 $\pi_{A,\boldsymbol{v}'}(\lambda)$ を用いて, アルゴリズムIの Step 2 を実行し, q 個の列ベクトル $\boldsymbol{w}_1^{\prime\prime(0)}, \boldsymbol{w}_2^{\prime\prime(0)}, ..., \boldsymbol{w}_q^{\prime\prime(0)}$ を求める.
- これらのベクトルを用いて, Step 3 と同様の計算を行うことで, 各 $j \in J'$ に関する基本最小消去多項式候補の計算を行う.

以上が, 第3節に与えた基本最小消去多項式候補を求める計算法の改良の概略である.

5 最小多項式計算への応用

この節では, 基本最小消去多項式候補の応用として, 行列の最小多項式を求める新たな計算法を提案する. 第4節に与えた計算法がその基本をなしている. 以下にその概略を与える.

- 行列 A の最小多項式候補を求める
- 基本最小消去多項式候補 $\pi'_{A,j}(\lambda), j = 1, 2, ..., n$ を求める
- 各 j に対し, $\pi'_{A,j}(A)\mathbf{e}_j$ を計算し, $\pi'_{A,j}(\lambda) = \pi_{A,j}(\lambda)$ か否か確かめる.
- $\pi'_{A,j}(\lambda) = \pi_{A,j}(\lambda)$, $j = 1, 2, ..., n$ なる場合, $\pi_A(\lambda) = \mathrm{lcm}\{\pi_{A,1}(\lambda), \pi_{A,2}(\lambda), ..., \pi_{A,n}(\lambda)\}$ と定める.

以上がその概略である.

常識的には「基本最小消去多項式 $\pi_{A,j}(\lambda), j = 1, 2, ..., n$ は最小多項式 $\pi_A(\lambda)$ より, 行列 A に関する多くの情報を含んでいるから, 基本最小消去多項式を求めてから最小多項式を決定するという方法は計算効率が良い訳が無い」と考えるのが普通である. しかし, 今, 最小多項式候補 $\pi'_A(\lambda)$ のみが与えられたとし, $\pi'_A(\lambda)$ が真の最小多項式であるか否か, $\pi'_A(A)$ を計算することで確かめるとする. その際の計算量は, $\pi'_A(\lambda) = \pi_A(\lambda)$ である場合 (通常のホーナー法を用いたとして) 掛け算の回数を行列・ベクトル積の掛け算の回数に換算すると $n \deg \pi_A$ 回となる. 他方, $\pi'_{A,j}(\lambda) = \pi_{A,j}(\lambda), j = 1, 2, ..., n$ か否かを確かめるのに必要な行列・ベクトル積の回数は $\sum_j \deg \pi_{A,j}$ 回である. 従って, 提案法では, 検算の際に

$$n \deg \pi_A - \sum_j \deg \pi_{A,j}$$

ほど行列・ベクトル積の回数が少ないことになる. また, 論文 [9] の３節で述べたように, これらの計算はすべて並列化することが可能である. しかも, 計算を分割する際, 並列化による効果がほぼ最適となるような分割を容易に与えることが出来るという特徴を持つことも重要である.

参考文献

[1] D. Augot, P. Camion : On the computation of minimal polynomials, cyclic vectors, and Frobenius forms, Linear. Alg. Appl. **260** (1997), 61–94.

[2] 飯塚由貴恵, 田島慎一 : 行列のスペクトル分解アルゴリズムについて – 最小多項式が複数の重複因子から成る場合 – , 京都大学数理解析研究所講究録掲載予定.

[3] 小原功任, 田島慎一 : 行列のスペクトル分解・固有ベクトルの分散計算, 京都大学数理解析研究所講究録, **1666** (2009), 65–68.

[4] 小原功任, 田島慎一 : 最小消去多項式を用いた行列スペクトル分解計算の並列化, 京都大学数理解析研究所講究録投稿中.

[5] 木村欣司 : http://www-is.amp.i.kyoto-u.ac.jp/kkimur/

[6] 田島慎一, 飯塚由貴恵 : 行列のスペクトル分解アルゴリズムについて, 京都大学数理解析研究所講究録, **1666** (2009), 49–56.

[7] 田島慎一, 樋口水紀：レゾルベントを用いた固有ベクトル計算, 京都大学数理解析研究所講究録, **1666** (2009), 57–64.

[8] 田島慎一, 奈良洸平, 小原功任：行列の最小多項式計算について, 京都大学数理解析研究所講究録掲載予定.

[9] 田島慎一：微分作用素を用いたレゾルベントの留数解析と行列のスペクトル分解, 京都大学数理解析研究所講究録掲載予定.

[10] 照井章, 田島慎一：行列の最小消去多項式候補を利用した固有ベクトル計算, 京都大学数理解析研究所講究録投稿中.